A Writing Assistent Using Language Models Derived From the Web

Sharon Tsai
d00yt@efd.lth.se

Abstract

In the field of Linguistic there exists many powerful tools for measuring the statistic characteristics of words and sentences. These tools rely on a corpus to which the data is compared. In order to get good and meaningful results from the tools available, a suitable corpus is thus needed. As the corpus is the key that ties the tools together, it is of uttermost importance. For most applications, all though not all, a large corpus is useful. This paper presents a solution to using the largest corpus known to man, the Internet. It will show a prototype program using many different linguistic tools on information gathered by the premiere search engine Google.

1 Method

The basic mathematical tools needed for this work is explained in this section. For a more in depth explanation please see (Language Processing Computational Linguistics, 2003).

1.1 N–Grams

N–Grams is simply a method of counting the frequency of a sequence of N word in the corpus. These frequencies can then be used to calculate probabilities. Equation 1 shows the probability of a single word occurring. It is naturally the frequency of the word occurring in the corpus divided by the amount of different words available in the corpus.

$$P(w) = \frac{C(w)}{N} \tag{1}$$

It is also possible to calculate probabilities of a word following a given word. This is known as the maximum likelihood estimate and is defined in equation 2.

$$P(w_i|w_{i-1}) = \frac{C(w_{i-1}, w_i)}{C(w_{i-1})} \tag{2}$$

The maximum likelihood estimate for a word following two given words is defined in equation 3.

$$P(w_i|w_{i-2}, w_{i-1}) = \frac{C(w_{i-2}, w_{i-1}, w_i)}{C(w_{i-2}, w_{i-1})} \tag{3}$$

1.2 Mutual Information

Mutual Information is a tool for measuring the strength of word associations. A high value is an indication of two words occurring together but with a total small frequency, such as technical terms. For instance, such terms might be ”hyperthreading processor” or ”keyhole surgery”. Mutual Information is defined in equation 4

$$I(w_i, w_{i+1}) = \log_2 \frac{NC(w_i, w_{i+1})}{C(w_i)C(w_{i+1})} \tag{4}$$

1.3 T–Score

T–Score is a statistic tool which measures frequently occurring grammatical combinations. A high T–Score means that the two words occur often together, such as ”of the” and ”in the”. The

definition of T–Score is shown by equation 5

$$T(w_i, w_{i+1}) = \frac{C(w_i, w_{i+1}) - \frac{1}{N}C(w_i)C(w_{i+1})}{\sqrt{C(w_i, w_{i+1})}} \quad (5)$$

2 Implementation

2.1 Module Overview

The prototype program is implemented in Java. For more information please see (J2SE 1.4.1 API Specification, 2003). It is made up by five main classes: GUI, UserPane, SearchResultPane, SearchHandler and HTMLWriter. See figure 1 for the overview of the structure.

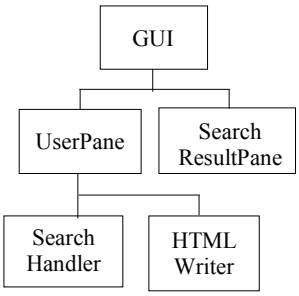

Figure 1: An overview of the architecture

2.2 User Interface

2.2.1 GUI

The GUI is a frame that contains a UserPane and a SearchResultPane. A user can change view by clicking on the respective tab that represent UserPane and SearchResultPane.

2.2.2 UserPane

The UserPane consists of a button group, a text field, a text area, two buttons, Send and Reset, and a status label. The button group contains a list of linguistic methods that a user can choose to analyze the input sentence with. Only one method can be chosen at the same time. These methods that are included in the prototype program are N-Gram, Mutual Information and T-Score. The text field is where the user types in the sentence which is to be analyzed. The final result is presented in the text area below the text field. The status label at the bottom updates the program status, search progress and possible error messages. See figure 2, 3 and 4.

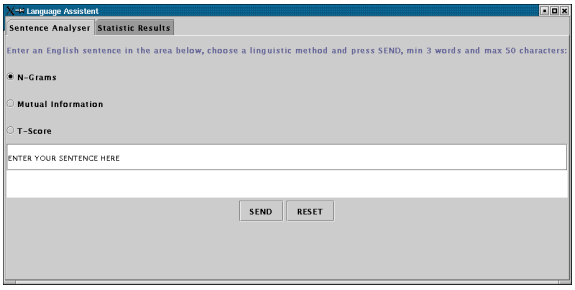

Figure 2: GUI: UserPane

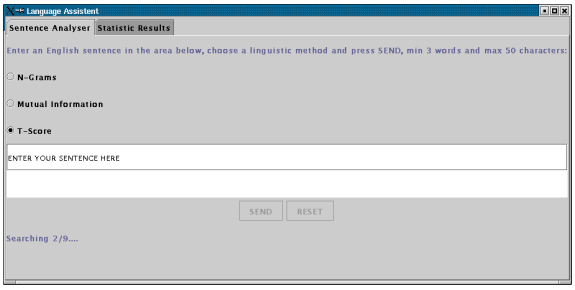

Figure 3: GUI: UserPane during search

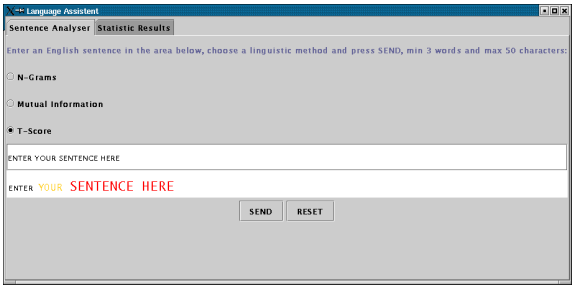

Figure 4: GUI: UserPane after executing search

2.2.3 SearchResultPane

The SearchResultPane consists of a web page that is generated automatically in the end of every search. See figure 5. The web page reloads automatically after every new search and contains all the statistical data gathered.

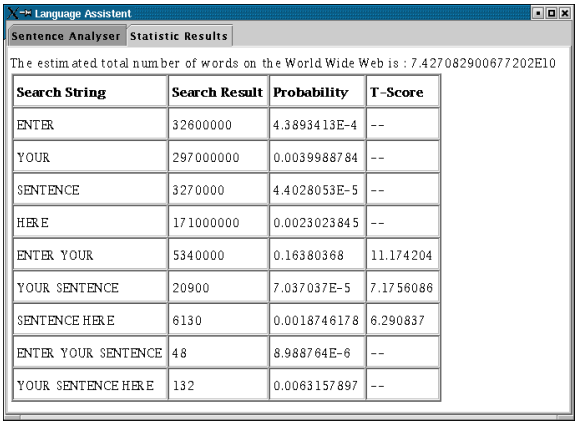

Figure 5: GUI: SearchResultPane

2.3 Implementation Detail

The data flow chart, figure 6, shows all seven stages of the prototype program. See the corresponding sections for further detail and information on each stage.

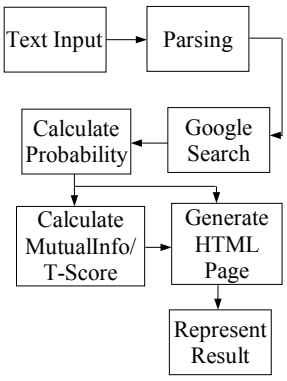

Figure 6: Flow chart

2.3.1 Initiating GUI

During the program initiation, the total number of words that exists on the Internet is estimated. This is done by calculating the occurring percentage of words such as ”in,” ”on,” and ”of” in a fixed size English corpus. This percentage is then applied on the sum of the Google search results of these words in order to estimate total number of words. The percentage used in this program is obtained through Jane Austin’s novel ”Emma.” There are around 74.6 billion words on the Internet and the number is growing everyday.

2.3.2 Text Input

This stage is the very beginning of the program cycle. It is done simply by clicking on the Send button or press ”ENTER” on keyboard after entering a sentence in the text field. At the moment, the size of the sentence is limited at 50 tokens. The sentence is then sent to the SearchHandler object.

2.3.3 Parsing

The SearchHandler first applies an error control on the incoming sentence. The main purpose of the error control procedure is to filter out the word ”the,” which causes overflow in the automated Google search. The SearchHandler also checks if the sentence is empty. If any of these two errors occurs, the program is terminated and an appropriate error message displays at the status label. The sentence is then parsed into unigrams, bigrams and trigrams if no errors are found. All N-Grams are stored in an array.

2.3.4 Google Search

The SearchHandler traverses the N-Gram array and performs Google search on every N-Gram. Google returns an estimated result back and it is saved in the corresponding position in a separate array for search results.

Google’s own API is used in this program because Google does not permit automated queries without an API account. Communication is performed via Simple Object Access Protocol(SOAP). For further information about Google API see (Google API, 2003).

2.3.5 Calculate Probability

No matter which linguistic method is chosen to analyze the input sentence, N-Gram probabilities are always calculated according to formulas presented in section 1. The resulting N-Gram probabilities are stored in a separate array in the corresponding position.

2.3.6 Calculate Mutual Information/ T-Score

If the chosen linguistic method to analyze the input sentence is Mutual Information or T-Score, values for bigrams are calculated according to formulas presented in section 1. These values are also stored in a separate array.

2.3.7 Generate HTML Page

After all the necessary values are calculated, a web page is generated by HTMLWriter object and loaded into the SearchResultPane. It can also be viewed by any web browser. The web page contains the estimated total number of words on the Internet, and a table of all N-Grams and their corresponding N-Gram probability and Information/ T-Score values, if any.

2.3.8 Represent Result

Depending on which linguistic method is chosen, the corresponding result array is traversed. Separate strategies are used to analyze results of N-Gram, Mutual Information and T-Score. There are two ways to represent unusual words, in orange style and red style. The font of words is slightly bigger than default, size sixteen, and painted in orange. In red style, it is size twenty font and painted in red.

For N-Gram, if any of the N-Gram probability is less than 0.1%, it is marked in orange style. If the N-Gram probability is less than 0.05%, it is marked in red style.

The bigram that has the highest Mutual Information value is marked in red style and the second highest bigram is marked in orange style.

The marking strategy for T-Score is the opposite for Mutual Information. The bigram that has the lowest T-Score value is marked in red style and the second lowest bigram is marked in orange style.

3 Experiments

3.1 Experiment I: ”I really like strawberry beer”

3.1.1 Idea

The sentence, ”I really like strawberry beer,” is used as reference sentence during the development phase because the very rare occurrance/use of the word combination ”strawberry” and ”beer.”

3.1.2 Search Results

As figure 7 clearly shows that all three methods manage to detect the unusual word combination, ”strawberry beer.” This is the primary reason that the Mutual Information marking strategy is chosen to mark the highest and the second highest value.

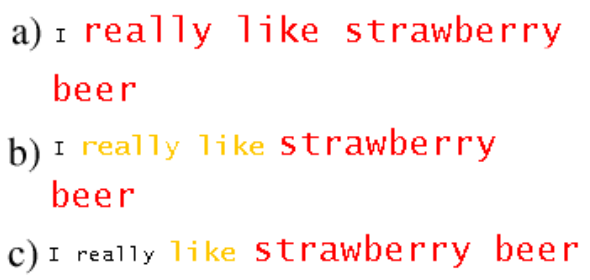

Figure 7: Search results: The sentence is analyzed by: a) N-Gram b) Mutual Information c) T-Score

3.1.3 Conclusion

In this example, the program demonstrates the ability to detect strange or unusual word combinations.

3.2 Experiment II: ”This project consists of three parts, and it consists at working”

3.2.1 Idea

For most people, the most difficult part of learning a foreign language is prepositions. In this example, the most common prepositions that occur with the word ”consist” are ”of” and ”in,” according to Oxford Advanced Learner’s Dictionary. The goal is that the program should be able to spot and mark the bigram, ”consists at” in red style, preferably, or in orange style because it is grammatically incorrect.

3.2.2 Search Results

Figure 8 shows that linguistic methods N-Gram and T-Score manages to marks the error bigram in red style, however, Mutual Information fails to produce correct results.

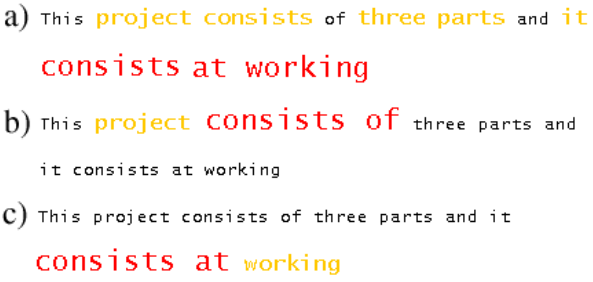

Figure 8: Search results: The sentence is analyzed by: a) N-Gram b) Mutual Information c) T-Score

With a closer look on the numerical values of the search result which presents in figure 9, the Mutual Information value of the error bigram is ranked the sixth highest or the fifth lowest of all ten bigrams which places it in the middle. Therefore, the error cannot be spotted by using Mutual Information method with the current strategy, whether by marking the bigram that has the highest or the lowest Mutual Information value.

Search String	Search Result	Probability	Mutual Information	
This project	2450000	0.0050619836	10.2722845	7
project consists	49600	0.0010530786	14.68117	2
consists of	1990000	0.41983122	18.298563	1
of three	3950000	0.0025816993	7.503225	8
three parts	384000	0.0074131275	14.622449	3
parts and	1900000	0.060126584	12.786431	4
and it	6590000	0.004393333	5.2354455	10
it consists	398000	9.191686E-4	11.28439	5
consists at	2550	5.379747E-4	10.633199	6
at working	49300	1.2386935E-4	5.7684116	9

Figure 9: Mutual Information values in SearchResultPane with additional rankings

3.2.3 Conclusion

This example demonstrates the program's ability to spot possible incorrect prepositions in a sentence. However, T-Score and N-Gram methods produce better and more reliable results than Mutual Information.

3.3 Experiment III: "He catch that ball after she drops it"

3.3.1 Idea

There is an obvious tense error in the sentence. It should be "He catches" instead of "He catch." With the help of any of these three linguistic methods, hopefully the program is able to detect this grammatic error.

3.3.2 Search Results

As figure 10 shows that all three methods manages to spot and mark the error bigram in some way. Again, T-Score produces the best results and is still the most powerful linguistic method compared to N-Gram and Mutual Information. Mutual Information only marks "catch" in orange style, not in red. It also misleadingly marks the bigram "drops it" in red style, which is grammatically correct.

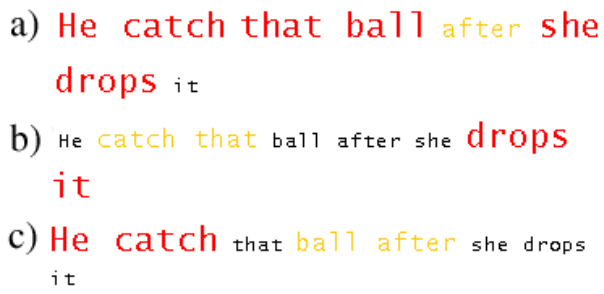

Figure 10: Search results: The sentence is analyzed by: a) N-Gram b) Mutual Information c) T-Score

3.3.3 Conclusion

The program also has the ability to analyze and check the verb tense in a sentence. However, it can only analyze a sentence at lexical and grammatic level, not in semantic level as it does not try to understand the meaning of the sentence.

4 Further work

There are a few further improvements that can be done in the program. Breaking up the option N-Gram into three different sub-options, unigram, bigram and trigram and during the parsing stage, the input sentence will only be parsed according to the linguistic methods, e.g. only bigrams in T-Score and bigram options and trigrams when the trigram option is selected. This might decrease the run time a little bit, but not very significantly.

If a user wishes to analyze the same sentence with different methods in different search, the program should be able to use the already existed search results from the previous search and calculate new values according to the selected method. This will decreases the run time significantly, especially after the first initial search.

More sophisticated marking strategies can be developed for all the methods instead of the existing straight-forward approach as in T-Score, the lowest in red style, and in N-Gram, under 0.05% marked in red style.

5 Conclusion

This paper has clearly showed that it is possible to implement a language assistance using linguistic methods without a fixed size corpus. The Internet is without doubt the biggest corpus ever created. The number of web pages on the Internet increases everyday. However, the difference in quality and the form and use of the language of these home pages are substantially large. Without carefully analyzing and filtering out the unwanted information, the results may be misleading. The search results can very well represent the most modern, normal-everyday-life, down-to-earth form of a language.

It has been a great learning experience and challenge in both applying textbook formulas into real-life uses and designing a user-friendly program. A few mistakes has been made and corrected, but the most important of all, is the satisfaction of enjoying the final fruit.

References

Pierre Nugues. 2003. *Compendium, Language Processing Computational Linguistics*

Sun Microsystems. 2003. *J2SE 1.4.1 API Specification* java.sun.com

Per Holm 2000 *Objektorienterad programmering och Java*

Oxford University Press 2003 *Oxford Advanced Learner's Dictionary* www.oup.com/elt/global/products/oald/

Google 2003 *Google API* www.google.com/apis